Paid $1

14

YITAI HU (SBN 248085)
yitai.hu@alston.com
SEAN P. DEBRUINE (SBN 168071)
sean.debruine@alston.com
ALSTON & BIRD LLP
Two Palo Alto Square
3000 El Camino Real, Suite 400
Palo Alto, California 94306-2112
Telephone: (650) 838-2000
Facsimile: (650) 838-2001

E-filing

Attorneys for Plaintiff
OURS TECHNOLOGY INC.

ADR

JL

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

OURS TECHNOLOGY, INC., a Taiwanese corporation,

Plaintiff,

vs.

DATA DRIVE THRU, INC., a Texas Corporation,

Defendants.

Case No. C09 00585 JL

COMPLAINT FOR DECLARATORY JUDGMENT

DEMAND FOR JURY TRIAL

ORIGINAL

Plaintiff Ours Technology, Inc. files this complaint against Data Drive Thru, Inc. and alleges as follows:

PARTIES AND BACKGROUND

1. Plaintiff Ours Technology, Inc. ("OTi") is a company organized under the laws of Taiwan, with a place of business at No. 85, Guangming 6th Road, Jhubei City, Hsinchu County 30268, Taiwan, R.O.C.

2. OTi is a leading solution provider for USB-based multimedia and transmission products, such as removable storage devices (including USB Flash Drives, Card Readers and Flash Cards), data transfer cables, and personal media devices.

3. OTi's employee, Shih-Chou Juan, invented an innovative method for USB file transfer between two computers via a USB cable including a self-enabling, built-in application program embedded in the data transfer cable itself ("Auto-run File Transfer Invention"). This invention obviated the need for users to install programs on either computer in order to perform file transfers. OTi filed a patent application for the Auto-run File Transfer Invention with the United States Patent and Trademark Office ("USPTO") on August 7, 2003. A true copy of the OTi's published patent application is attached as Exhibit A.

4. Upon information and belief, Defendant Data Drive Thru, Inc. ("DDT") is a Texas corporation with a place of business at 100 Crescent Court, Suite 700, Dallas, Texas 75201, and is doing business in this judicial district.

5. DDT is the assignee of United States Patent No. 7,108,191 ("the '191 Patent"). The '191 Patent was filed on October 19, 2004 and is directed to a technology that is the same, or an obvious variation of, OTi's Auto-run File Transfer Invention. A true copy of the '191 Patent is attached as Exhibit B.

6. DDT is in the business of making, importing, selling, offering to sell, marketing and/or distributing, directly or through its authorized agents and distributors, data transfer devices. Upon information and belief, these data transfer devices are being sold or offered for sale at over 3,500 retail locations nationwide, including in this jurisdiction. In addition, DDT operates a web site through which it offers its products for sale to residents of this jurisdiction

and, on information and belief, sells its products directly to residents of this jurisdiction. Through these and other actions, DDT has continuous and systematic contact with this jurisdiction. DDT has purchased ICs designed and sold by OTi for purposes of incorporating OTi's products into DDT's data transfer devices.

NATURE OF THE ACTION

7. This is an action for declaratory judgment of non-infringement and invalidity of the '191 Patent pursuant to the Declaratory Judgment Act, 28 U.S.C. §§ 2201-02, and the Patent Laws of the United States, 35 U.S.C. §§ 100, *et seq.*, and for such other relief as the Court deems just and proper.

JURISDICTION AND VENUE

8. On December 30, 2008, DDT filed suit in the United States District Court for the Eastern District of Texas ("Texas Suit") against RadioShack Corporation, Targus, Inc., Samsung Electronics America, Inc., and ACCO Brands Corporation, alleging infringement of its '191 Patent. On information and belief, prior to filing that lawsuit DDT directed correspondence and/or other communications alleging infringement and demanding payment to Targus, Inc. in California.

9. Each of the named defendants in the Texas Suit is, or has been, a customer for OTi's products that implement OTi's Auto-run File Transfer Invention. The Texas Suit is directed to OTi's Auto-run File Transfer feature.

10. Plaintiff OTi has applied its cutting-edge Auto-run File Transfer Invention to design products and solutions directed to data transfer services. OTi's products and solutions have not infringed and do not infringe, either directly or indirectly, any valid and enforceable claim of the '191 Patent.

11. Therefore, a substantial controversy exists between the parties concerning the validity, enforceability and/or infringement of the '191 Patent, which is of sufficient immediacy and reality to warrant declaratory relief.

12. This Court has jurisdiction over the subject matter of this action pursuant to 28 U.S.C. §§ 1331 and 1338.

13. Venue is proper in this district pursuant to 28 U.S.C. §§ 1391 and 1400.

INTRADISTRICT ASSIGNMENT

14. This is an Intellectual Property Action and shall therefore be assigned on a district wide basis in accordance with Local Rule 3-2(c).

COUNT I

DECLARATORY JUDGMENT OF NON-INFRINGEMENT OF THE '191 PATENT

15. OTi incorporates by reference the allegations set forth in the preceding paragraphs as if fully set forth herein.

16. OTi does not infringe directly, contributorily or by inducement any valid and enforceable claim of the '191 Patent.

COUNT II

DECLARATORY JUDGMENT OF INVALIDITY OF THE '191 PATENT

17. OTi incorporates by reference the allegations set forth in the preceding paragraphs as if fully set forth herein.

18. The '191 Patent is invalid for failure to satisfy the conditions of patentability specified under 35 U.S.C. §§ 102, 103, and/or 112. By way of example and without limitation, the '191 Patent is invalid under 35 U.S.C. § 102(g) over at least OTi's Auto-run File Transfer Invention.

PRAYER FOR RELIEF

WHEREFORE, OTi prays for judgment against Defendant DDT and an order:

A. Declaring that OTi has not infringed any valid and enforceable claim of the '191 Patent;

B. Declaring that the claims of '191 Patent are invalid;

C. Preliminarily and permanently enjoining DDT, its officers, directors, servants, managers, employees, agents, successors and assignees, and all persons in active concert or participation with any of them from directly or indirectly charging OTi with infringement of the '191 Patent under any theory;

D. Declaring that this is an exceptional case under 35 U.S.C. 285;

E. Awarding OTi's costs, expenses, and attorney's fees in this action; and

F. Granting such other and further relief as the Court may deem appropriate.

Dated: February 9, 2009

Respectfully submitted,

ALSTON & BIRD LLP

By _____
Sean P. DeBruine

Attorneys for Plaintiff
OURS TECHNOLOGY INC.

DEMAND FOR JURY TRIAL

Pursuant to Federal Rule of Civil Procedure 38(b), Ours Technology Inc. respectfully demands a trial by jury on all issues so triable.

Dated: February 9, 2009

Respectfully submitted,

ALSTON & BIRD LLP

By _____
Sean P. DeBruine

Attorneys for Plaintiff
OURS TECHNOLOGY INC.

CERTIFICATION OF INTERESTED ENTITIES OR PERSONS

Pursuant to Civil L.R. 3-16, the undersigned discloses that Ours Technology Inc. is not aware of any other non-party interested entities or persons.

Dated: February 9, 2009

Respectfully submitted,

ALSTON & BIRD LLP

By _____
Sean P. DeBruine

Attorneys Plaintiff
OURS TECHNOLOGY INC.

EXHIBIT A

US 20040230708A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2004/0230708 A1**

Juan (43) **Pub. Date: Nov. 18, 2004**

(54) **APPLICATION METHOD FOR UNIVERSAL SERIAL BUS FILE TRANSFER CABLE**

(75) Inventor: **Shih-Chou Juan**, Jungli City (TW)

Correspondence Address:
TROXELL LAW OFFICE PLLC
SUITE 1404
5205 LEESBURG PIKE
FALLS CHURCH, VA 22041 (US)

(73) Assignee: **Ours Technology Inc.**

(21) Appl. No.: **10/635,549**

(22) Filed: **Aug. 7, 2003**

(30) **Foreign Application Priority Data**

May 13, 2003 (TW)..... 092113004

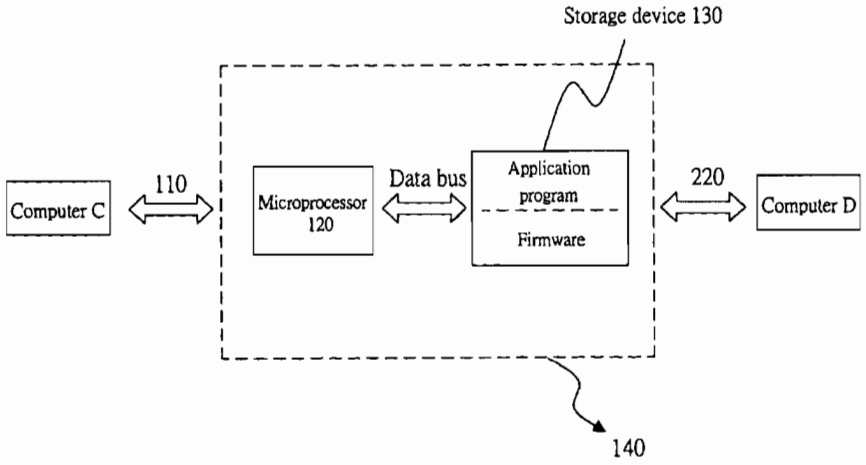
Publication Classification

(51) **Int. Cl.**[7] **G06F 3/00**

(52) **U.S. Cl.** **710/1**

(57) **ABSTRACT**

The present invention discloses an application method for universal serial bus (USB) file transfer cable. When USB file transfer cable is connected between two computers loaded with operating such systems as Window, Mac, Linux, etc., that support Mass Storage Class standard while the USB file transfer cable also supports Mass Storage Class, either computer can access system information provided in the USB file transfer cable. Since the application program is stored in the USB file transfer cable and the USB file transfer cable is simulated as an auto-run storage device, such as a floppy disc drive, a hard disc drive or a CD-ROM drive, etc., the USB file transfer cable in the present invention is capable of carrying out file transfers automatically without having to install drivers and programs. To users, the application method for USB file transfer cable offers plug and play capabilities as file transfers can be performed automatically between two computers without having to install drivers and programs in either computer.



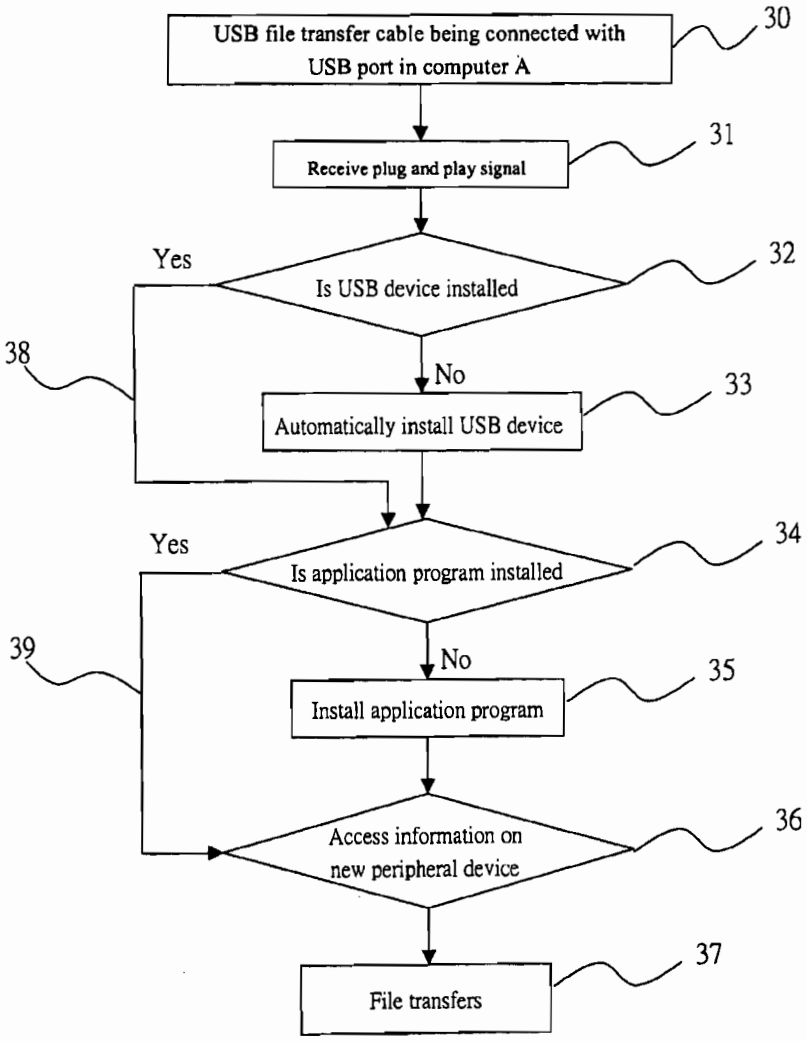


FIG. 1

(PRIOR ART)

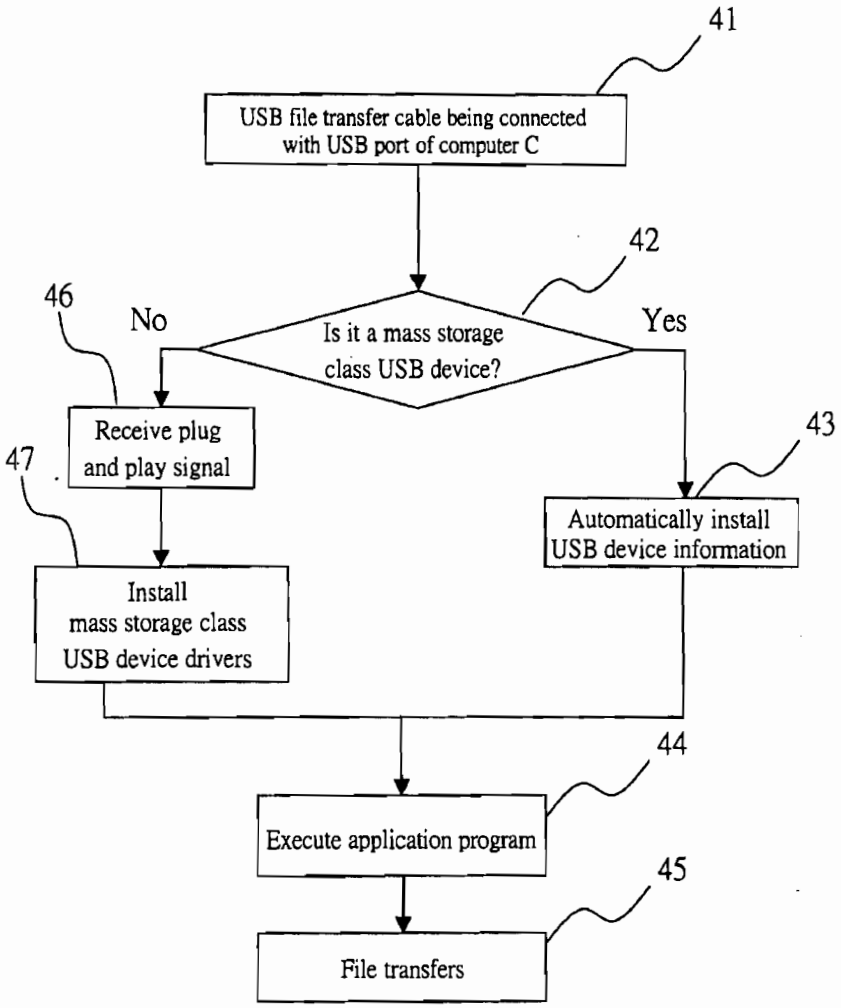

41
USB file transfer cable being connected with USB port of computer C
42
Is it a mass storage class USB device?
No
Yes
46
Receive plug and play signal
47
Install mass storage class USB device drivers
43
Automatically install USB device information
44
Execute application program
45
File transfers

FIG. 2

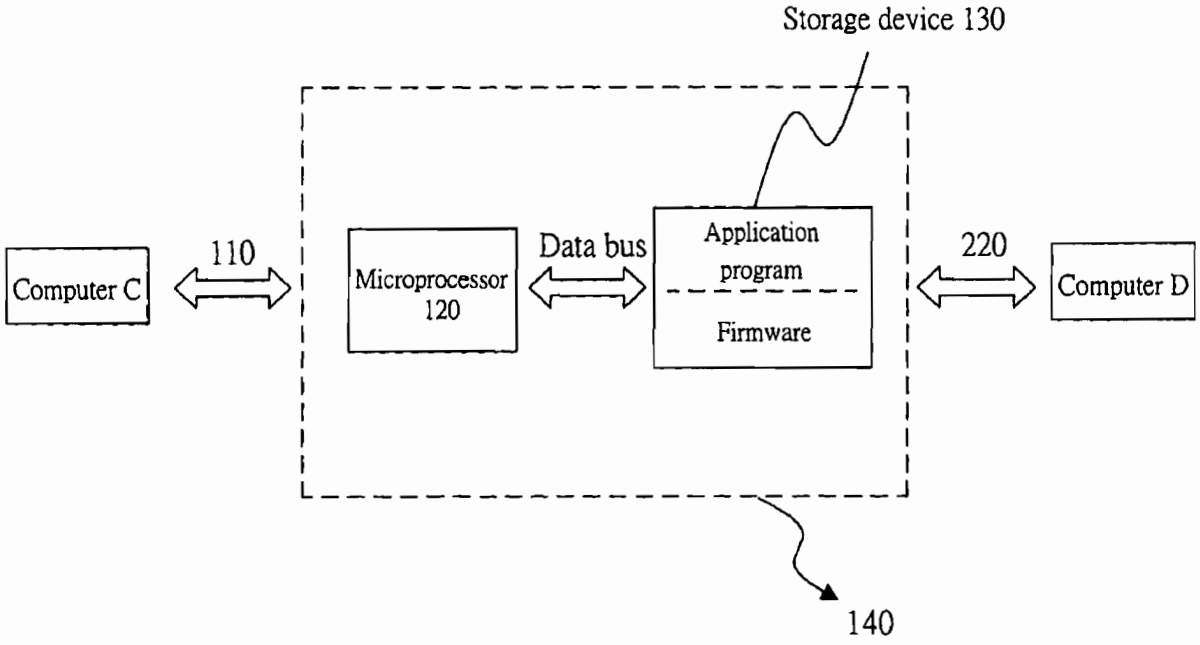

Storage device 130
Computer C
110
Microprocessor 120
Data bus
Application program
Firmware
220
Computer D
140

FIG. 3

APPLICATION METHOD FOR UNIVERSAL SERIAL BUS FILE TRANSFER CABLE

FIELD OF INVENTION

[0001] The present invention relates to an application method for universal serial bus (hereafter USB) file transfer cable and, in particular, relates to one that, on a mass storage class platform, transfers files between computers without having to install programs.

BACKGROUND OF INVENTION

[0002] Normally, data transmission between two computers can be achieved via the application of floppy disks or storage devices (CD-ROM, MO or Zip), direct cable connection cable (LL3/LI5 file transfer cable), LAN (local area network) and USB file transfer cable, etc.

[0003] The disadvantage of using floppy discs is the small storage space and slow transmission rate. A floppy disc provides the memory capacity of 1.44 MB. When transferring files greater than 1.44 MB, one has to transfer files in multiple diskettes, often with the aid of multiple compressions. When transferring large amount of data via storage devices such as CD-ROM, MO or ZIP, files in computer A have to be stored on a storage media (CD, MO or ZIP) by a storage device (CD-RW drive, MO or ZIP) connected with computer A before being copied over to computer B by a storage device (CD-ROM drive, MO or ZIP) connected with computer B. This process of course is inefficient.

[0004] Using LL3/LL5 file transfer cable has the advantage of not having to transfer through a storage media. However, the drawbacks are three-fold: slow transmission speed (230 Kbps), occupation of parallel port and short effective transmission distance. Although it is convenient to utilize network cable via LAN for data transmission, there is limitation to the environment that it can operate. Data transmission between computers requires the link-up of network card and network cable. Unlike office users, home users are usually not equipped with LAN environment, unless the family has more than three computers, wherein a LAN-connected area can then be set up. Nowadays, it is common for families to own both notebook and desktop computer, and it would be difficult if sizable data transfers between two computers are not carried out via LAN. This problem can be overcome by prior art application of USB file transfer cable. The prior art application method for USB file transfer cable requires the installation of drivers in both computers in order to facilitate data transfer. This would not pose much problem in domestic use, as it is not inconvenient for a user to install drivers on both computers. When a visitor (sales personnel or application engineer) pays a visit to a customer and wishes to transfer large amounts of data, both computers (visitor's and customer's) must first be installed with drivers before utilizing prior art application method for USB file transfer cable. This often becomes troublesome for both parties, thus its practicality being greatly discounted. Hence, the equipment vendors have a job cut out for them and that is to come up with solutions for resolving this problem.

[0005] We first describe the procedure of installing drivers when utilizing prior art application method for USB file transfer cable. Most USB file transfer cables support NDIS environment. **FIG. 1** illustrates a scenario, which begins with a USB file transfer cable being connected with the USB port in computer A (See step **30**), followed by the Windows 2000 operating system in computer A receiving a plug and play signal (See step **31**).

[0006] If the operating system is not loaded with a USB file transfer cable driver (See step **32**), a query window will appear inquiring a user whether to proceed with the installation of the driver following the insertion of manufacturer's CD. Unless a driver has already been installed, the user will install the driver before a USB device is established (See step **33**).

[0007] Suppose a driver has already been installed, the USB device will be detected by the operating system and no further installation will be required (See step **38**). Then upon inserting manufacturer's CD, a query window will appear inquiring whether an application program has been installed (See Step **34**). Unless an application program has already been installed, the user will install the application program as required (See step **35**). Suppose an application program has already been installed, no further installation will be required (See step **39**). Rebooting both computers following installations, either computer can now access information on the new peripheral device (USB file transfer cable) (See step **36**) and enable the application programs for file transfers (See step **37**). It is noted that both computers have to be installed with a driver and an application program according to steps shown in **FIG. 1** before utilizing prior art application method for USB file transfer cable. In other words, computer B wanting to transfer files with the computer A has to undergo the same installation procedures as shown in the flowchart. That means when a visitor wishes to transfer sizable amounts of data between his and customer's computer, he must always bring with him drivers and application programs provided by prior art USB file transfer cable manufacturer and carry out the installation of the driver and application program before utilizing prior art application method for USB file transfer cable.

[0008] Aiming to improve prior art USB file transfer cable as described above, the present invention discloses a novel USB file transfer cable, capable of supporting the Mass Storage Class standard that stores the driver and application program within itself in such a way that the USB file transfer cable is simulated as an auto-run storage device, such as a floppy disc drive, a hard disk drive or a CD-ROM drive. Hence, the USB file transfer cable in the present invention is capable of carrying out file transfers automatically without having to install drivers and programs. To users, the application method for USB file transfer cable in the present invention offers plug and play capabilities as file transfers can be performed automatically between two computers without having to install drivers and programs in either computer.

SUMMARY OF INVENTION

[0009] The main objective of the present invention is to provide an application method for USB file transfer cable wherein file transfers can be performed automatically by self-enabling its built-in application program without having to install drivers and programs in either computer.

[0010] Another objective of the present invention is to provide an application method for USB file transfer cable wherein, like the USB file transfer cable, operating systems

such as Windows, Mac, Linux, etc., support the Mass Storage Class standard. Thus, computers loaded with any of these operating systems can automatically access system information provided in the USB file transfer cable and take control of the operation of the peripheral device (i.e., USB file transfer cable) without having to install drivers and programs in either computer.

[0011] A further objective of the present invention is to provide an application method for USB file transfer cable wherein the USB file transfer cable can be simulated as an auto-run storage device, such as a floppy disc drive, a hard disc drive or a CD-ROM drive, etc. Thus, file transfer can be performed automatically by self-enabling its built-in application program without having to install drivers and programs in either computer.

[0012] First, computer C equipped with the driver of the standard USB Mass Storage Class can automatically access system information provided in the USB file transfer cable and take control of the operation of the peripheral device (i.e., USB file transfer cable) such that the operating system loaded in computer C can self-install the driver of the standard USB Mass Storage Class without having to install drivers and programs in computer C.

[0013] Likewise, when the USB file transfer cable is connected to a USB port of computer D, computer D, which is equipped with the driver of the standard USB Mass Storage Class, can automatically access system information provided in the USB file transfer cable and take control of the operation of the peripheral device (i.e., USB file transfer cable) such that the operating system loaded in computer D can self-install the driver of the standard USB Mass Storage Class without having to install drivers and programs in computer D.

[0014] Since drivers have been installed in computer C and D, the USB file transfer cable can be simulated as an auto-run storage device, such as a floppy disc drive, a hard disc drive or a CD-ROM drive, etc. File transfers can thus be carried out automatically by self-enabling its built-in application program so that users do not have to install drivers and programs in computer C and computer D. Therefore, the application method for USB file transfer cable in the present invention offers plug and play capabilities as file transfers can be performed automatically between two computers without having to install drivers and programs in either computer.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

[0015] The present invention aims to provide plug and play solutions for resolving problems of utilizing prior art application method for USB file transfer cable.

[0016] Capable of supporting the Mass Storage Class standard, the USB file transfer cable disclosed in the present invention stores the driver and application program within itself in such a way that the USB file transfer cable is simulated as an auto-run storage device, such as a floppy disc drive, a hard disk drive or a CD-ROM drive. Like the USB file transfer cable, operating systems such as Windows, Mac, Linux, etc., support the Mass Storage Class standard. Thus, computers loaded with any of the operating systems are equipped with standard storage driver in compliance with the USB Mass Storage Class and can automatically access system information provided in the USB file transfer cable and take control of the operation of the peripheral device (i.e., USB file transfer cable). Being simulated as an auto-run storage device, such as a floppy disc drive, a hard disk drive or a CD-ROM drive, etc., the USB file transfer cable in the present invention is capable of carrying out file transfers automatically without having to install drivers and programs. To users, the application method for USB file transfer cable in the present invention offers plug and play capabilities as file transfers can be performed automatically between two computers without having to install drivers and programs in either computer.

[0017] Flowcharts are being provided in order to illustrate the procedures in accordance with the present invention. Supporting the Mass Storage Class standard, the USB file transfer cable in the present invention carries out file transfers between computer by utilizing its built-in application programs and the default drivers in the operating system. If computer C (the master) as shown in **FIG. 2** utilizes Microsoft Windows 2000 operating system, Mass Storage Class is provided as a common standard. When a USB file transfer cable in the present invention is connected with a USB port of computer C (See step **41**), the operating system in computer C will automatically detect whether it is a Mass Storage Class USB device (See step **42**). Being a Mass Storage Class USB device, the USB file transfer cable in the present invention is able to utilize default USB drivers provided in Windows 2000 as drivers for its USB file transfer cable.

[0018] Users will be informed by the operating system of the existence of a USB file transfer cable so that re-installation is not required and Windows 2000 can access information in the USB file transfer cable. Taking only seconds to complete, driver installation and program execution are carried out automatically without further instructions.

[0019] When the USB file transfer cable device is connected with computer D (the remote), application program in USB file transfer cable will be automatically executed, thereby facilitating file transfers (See step **45**).

[0020] Similarly, when the USB file transfer cable device is connected with a USB port of computer D (the remote), the USB file transfer cable will automatically install drivers in computer D (See steps **41**→**42**→**43**). The above embodiment illustrates the installation procedure under Windows 2000 operating system between the computers C and D. Application of the present invention includes such operating systems as Windows, Mac and Linux, etc. so long as these operating systems have Mass Storage Class as a common standard. Moreover, the present invention is not limited to computers loaded with those operating systems listed above. Any electronic data storage apparatus loaded with any of the operating systems that support Mass Storage Class, can benefit from the application method disclosed in the present invention.

[0021] For those operating systems that do not support Mass Storage Class (e.g., Windows 98 operating system), extra steps are required to manually install the driver. We now illustrate the installation procedures in a scenario when neither of the operating systems, connected by the USB file transfer cable according to the present invention, supports

Mass Storage Class. The installation procedures also apply when either of the operating systems does not support Master Storage Class. The installation procedures are as follows. When an USB file transfer cable connects with the USB port of computer C, the master, (See step **41**), the operating system will detect whether it is a Mass Storage Class USB device (See step **42**). Not having Mass Storage Class as a common standard, Windows 98 only detects a USB device and a plug & play signal is promptly displayed (See step **46**). Installation of USB device drivers (see step **47**) by a storage medium (such as a CD) is required in order to drive the USB file transfer cable as set forth in the embodiment. Suppose a USB file transfer cable has been installed, the operating system will inform users and as such, no further installation will be required.

[0022] Similarly, when the USB file transfer cable device is connected with a USB port of computer D (the remote), the USB file transfer cable will automatically install drivers in computer D (See steps **41**→**42**→**46**→**47**). Once the USB file transfer cable device is installed between computer C and D, application program in USB file transfer cable will be automatically executed (See step **44**), thereby facilitating file transfers (See step **45**).

[0023] As shown in **FIG. 3**, the main inventiveness in the present invention is providing a controller **140**, having a storage device **130** therein, in USB file transfer cables such that suitable application program and firmware, both supporting Mass Storage Class, are being provided in the storage device **130**. As such, the USB file transfer cable can utilize the default drivers in the operating system. Since the USB file transfer cable is simulated as an auto-run storage device, such as a floppy disc drive, a hard disk drive or a CD-ROM drive, having built-in auto-run file like auto.run, the USB file transfer cable can automatically execute application programs. That means users can execute application programs and carry out file transfers without bothering with installation procedures. **FIG. 3** is the functional block diagram for a USB file transfer cable controller system, wherein computer C and D are connected with a controller **140** via cable **110** and **220**, respectively. The controller **140** comprises a microprocessor **120** and a storage device **130**. Comprising an ALU (arithmetic logical unit), a device controller, etc. (not shown), the microprocessor **120** reads and accesses information being transferred via cable **110**/**220**. Being a memory for temporarily storing application program and firmware, the storage device **130** can be one of the four types of memory (flash memory, EPROM, EEPROM and ROM [MASK ROM]), or any two of those four types of memory. It can be the combination of a flash memory, an EEPROM and ROM or a flash memory, an EPROM and a ROM. Different methods are used to program different types of memory. Software is normally used to control flash memory while burning is used to program EPROM or EEPROM. In a preferred embodiment of the present invention, application program and firmware are built-ins in the memory as mentioned above. Therefore, file transfers between two computers, being linked according to the present invention, are not different from those between two computers in a LAN environment. By eliminating manual installation procedures associated with prior art application method; the present invention offers the benefit of utilizing USB file transfer cables as long as operating systems involved support Mass Storage Class. Taking only seconds to complete, driver installation and program execution in the present invention are carried out automatically without manual installation procedures.

[0024] Since USB file transfer cable offers such advantages as compatibility, high speed, convenience and unlimited capacity, all current computer systems support USB. USB 2.0 standard currently available offers 480 Mbps (bit per sec), thus suitable for transferring sizable amount of information. As a matter of fact, USB has become the most common interface among modern electronic storage devices. Being simulated as an auto-run storage device (floppy disc drive, hard disc drive and CD-ROM drive etc.), the USB file transfer cable in the present invention enables files transfers to be carried out automatically by self-enabling its built-in application program without having to install drivers and programs in either computer. To users, the application method for USB file transfer cable in the present invention offers plug and play capabilities as file transfers can be performed automatically between two computers without having to install drivers and programs in either computer. In the wake of the present invention, notebooks can be regarded as a removable disk with immense storage capacity.

[0025] The above embodiments are intended for describing the present invention without limiting the scope that the present invention may be applied. Modifications made in accordance with the disclosures of the present invention without departing from the spirits of the present invention are covered by the equivalents of the present invention.

BRIEF DESCRIPTION OF DRAWINGS

[0026] **FIG. 1** is a flowchart for the installation of prior art USB file transfer cable device between computer A and computer B.

[0027] **FIG. 2** is a flowchart for the installation of USB file transfer cable device between computer C and computer D according to the present invention.

[0028] **FIG. 3** is a flowchart for USB file transfer cable controller system according to the present invention.

What is claimed is:

1. An application method for USB file transfer cable being applied between a first and a second operating systems, both supporting Mass Storage Class standard while the USB file transfer cable also supporting Mass Storage Class standard, comprising the steps of:

a. the first and the second operating system automatically set up information for USB file transfer cable device by using pre-loaded drivers;

b. an application program provided in the USB file transfer cable is automatically executed in the first and the second operating system to set up information for the USB file transfer cable;

c. file transfers are being carried out.

2. The application method for USB file transfer cable of claim 1, wherein the application program in step b is a storage device provided in the USB file transfer cable.

3. The application method for USB file transfer cable of claim 2, wherein the storage device is one choosing from the group consisting a flash memory, an EPROM, an EEPROM and a MASK ROM.

4. The application method for USB file transfer cable of claim 2, wherein the storage device is the combination of any two choosing from the group consisting a flash memory, an EPROM and a MASK ROM.

5. The application method for USB file transfer cable of claim 2, wherein the storage device is the combination of any two choosing from the group consisting of a flash memory, an EEPROM and an MASK ROM.

6. The application method for USB file transfer cable of claim 2, wherein the storage device is the combination of a flash memory, an EPROM and a MASK ROM.

7. The application method for USB file transfer cable of claim 2, wherein the storage device is the combination of a flash memory, an EEPROM and a MASK ROM.

8. An application method for USB file transfer cable being applied between the first and the second operating system, neither supporting Mass Storage Class standard while the USB file transfer cable supporting Mass Storage Class standard, comprising the steps of:

a. an application program is executed in the first and the second operating system, respectively, to set up information for the USB file transfer cable;

b. an application program being provided in the USB file transfer cable is automatically executed in the first and the second operating system;

c. file transfers are being carried out.

9. The application method for USB file transfer cable of claim 8, wherein the application program in step b is a storage device provided in the USB file transfer cable.

10. The application method for USB file transfer cable of claim 9, wherein the storage device is one choosing from the group consisting a flash memory, an EPROM, an EEPROM and a MASK ROM.

11. The application method for USB file transfer cable of claim 9, wherein the storage device is combination of any two choosing from the group consisting a flash memory, an EPROM, an EEPROM and a MASK ROM.

12. The application method for USB file transfer cable of claim 9, wherein the storage device is the combination of a flash memory, an EPROM and a MASK ROM.

13. The application method for USB file transfer cable of claim 9, wherein the storage device is the combination of a flash memory, an EEPROM, and a MASK ROM.

14. An application method for USB file transfer cable, being applied between the first and the second operating system, the first operating system not supporting Mass Storage Class standard while the second operating system and the USB file transfer cable supporting Mass Storage Class standard, comprising the steps of:

a. to set up information for the USB file transfer cable, an application program is executed in the first operating system and an application program provided in the USB file transfer cable is automatically executed in the second operating system;

b. an application program being provided in the USB file transfer cable is automatically executed in the first and the second operating system;

c. file transfers are being carried out.

15. The application method for USB file transfer cable of claim 14, wherein step b is a storage device provided in the USB file transfer cable.

16. The application method for USB file transfer cable of claim 15, wherein the storage device is one choosing from the group consisting a flash memory, an EPROM, an EEPROM and a MASK ROM.

17. The application method for USB file transfer cable of claim 15, wherein the storage device is the combination of any two choosing from the group consisting a flash memory, an EPROM, an EEPROM and a MASK ROM.

18. The application method for USB file transfer cable of claim 15, wherein the storage device is the combination of a flash memory, an EPROM and a MASK ROM.

19. The application method for USB file transfer cable of claim 15, wherein the storage device is the combination of a flash memory, an EEPROM, and a MASK ROM.

20. An USB file transfer cable capable of supporting a Mass Storage Class standard, comprising:

a controller that includes a processor and a storage device, the storage device providing a built-in application program for the USB file transfer cable; and

two USB devices, each being connected with the controller and one of two computers via a cable.

21. The application method for USB file transfer cable of claim 20, wherein the storage device is one choosing from the group consisting a flash memory, an EPROM, an EEPROM and a MASK ROM.

22. The application method for USB file transfer cable of claim 20, wherein the storage device is the combination of any two choosing from the group consisting a flash memory, an EPROM, an EEPROM and a MASK ROM.

23. The application method for USB file transfer cable of claim 20, wherein the storage device is the combination of a flash memory, an EEPROM, and a MASK ROM.

24. The application method for USB file transfer cable of claim 20, wherein the storage device is the combination of a flash memory, an EEPROM, and on MASK ROM.

25. A USB file transfer cable having the characteristic:

the USB file transfer cable being simulated as an auto-run storage device such that file transfers can be carried out by automatically executing built-in application programs without having to install application programs.

* * * * *

EXHIBIT B

US007108191B2

(12) **United States Patent**
Andrus

(10) **Patent No.: US 7,108,191 B2**
(45) **Date of Patent: Sep. 19, 2006**

(54) **INTELLIGENT COMPUTER CABLING**

(75) Inventor: **Paul L. Andrus**, McKinney, TX (US)

(73) Assignee: **Data Drive Thru, Inc.**, McKinney, TX (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **10/968,056**

(22) Filed: **Oct. 19, 2004**

(65) **Prior Publication Data**

US 2006/0081716 A1 Apr. 20, 2006

(51) **Int. Cl.**
G06K 19/06 (2006.01)

(52) **U.S. Cl.** **235/492**; 235/487

(58) **Field of Classification Search** 235/492, 235/441, 486–487; 709/227, 250, 72; 710/72, 710/100

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,321,817	A *	6/1994	Feinstein	703/25
5,615,344	A *	3/1997	Corder	710/62
6,128,673	A *	10/2000	Aronson et al.	710/22
6,131,125	A *	10/2000	Rostoker et al.	709/250
6,256,688	B1 *	7/2001	Suetaka et al.	710/73
6,256,690	B1 *	7/2001	Carper	710/301
6,370,603	B1 *	4/2002	Silverman et al.	710/72
2003/0218550	A1 *	11/2003	Herrmann	340/870.39

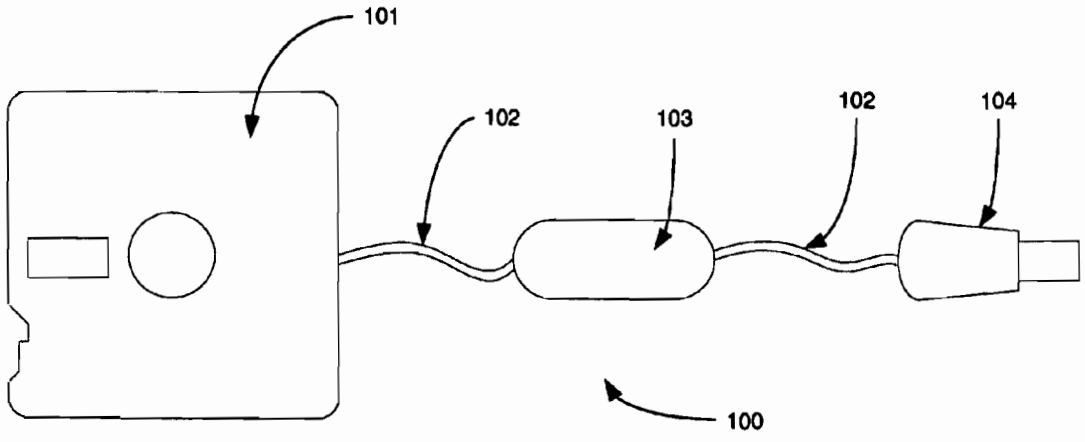
* cited by examiner

Primary Examiner—Thien M. Le
Assistant Examiner—Edwyn Labaze
(74) *Attorney, Agent, or Firm*—David W. Carstens; Carstens & Cahoon, LLP

(57) **ABSTRACT**

The present invention provides a data transfer system apparatus that automatically loads the necessary drivers and code through two interface elements to facilitate the transfer of electronic data from one EDP to another. In a preferred embodiment of the present invention the apparatus consists of a cable, USB interface plug, floppy drive transfer device (diskette), processor, controller, memory, circuitry components and software code. The electronic components and software applications are contained in the cable housing unit. Each interface element is attached at one end of the cable so they can be inserted into the respective EDP interfaces. Insertion of the apparatus into the EDP interfaces automatically triggers the execution of the embedded software for auto loading of the necessary code to control the transfer of the data directly from one EDP to the other. The system emulates the apparatus as a peripheral device attached to the EDP through its USB port interface coupled to the other EDP using its FDD to transfer the data, using the data storage capacity of the receiving EDP as the serial bus end-point.

15 Claims, 9 Drawing Sheets



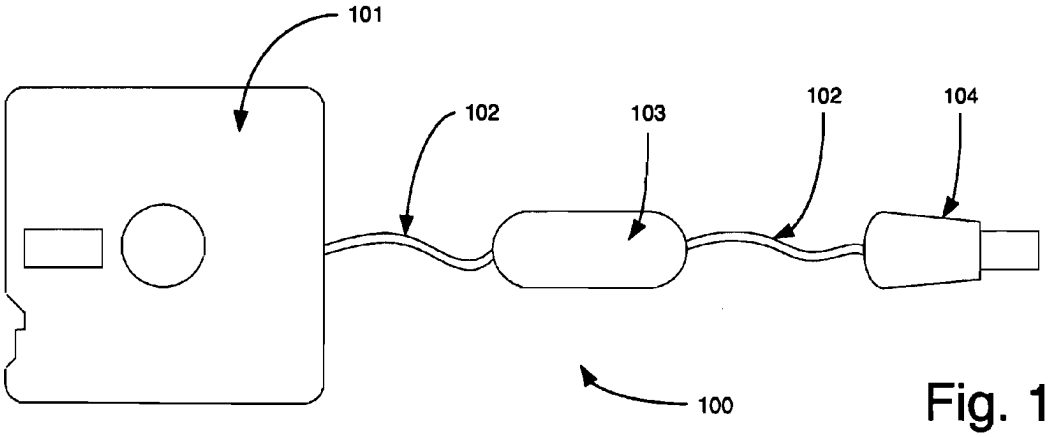


Fig. 1

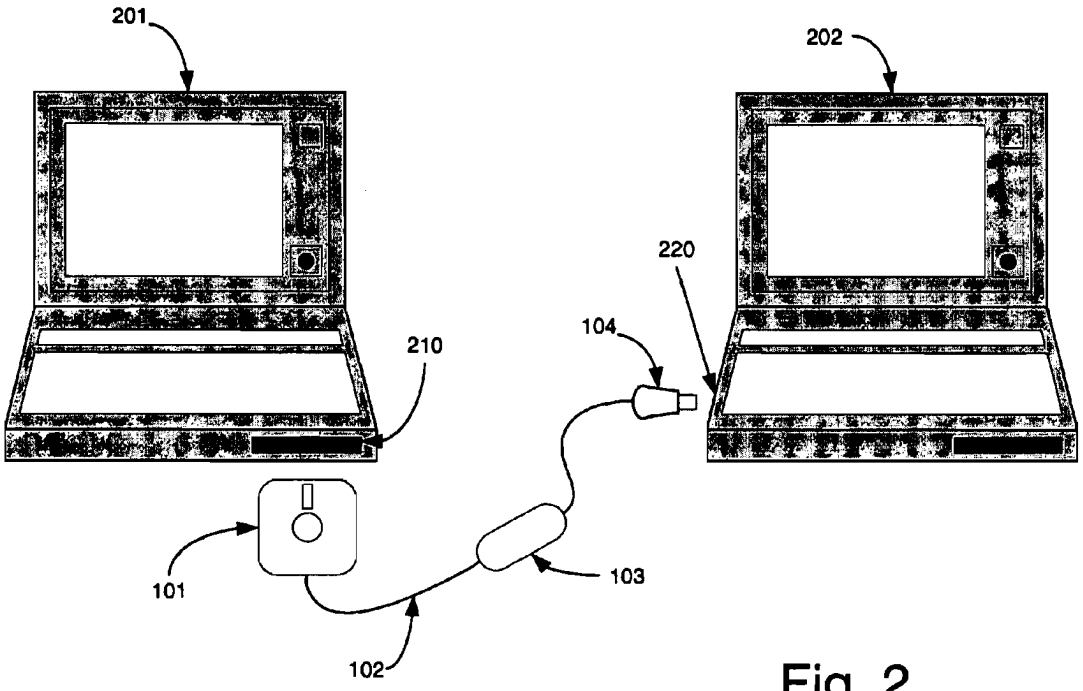


Fig. 2

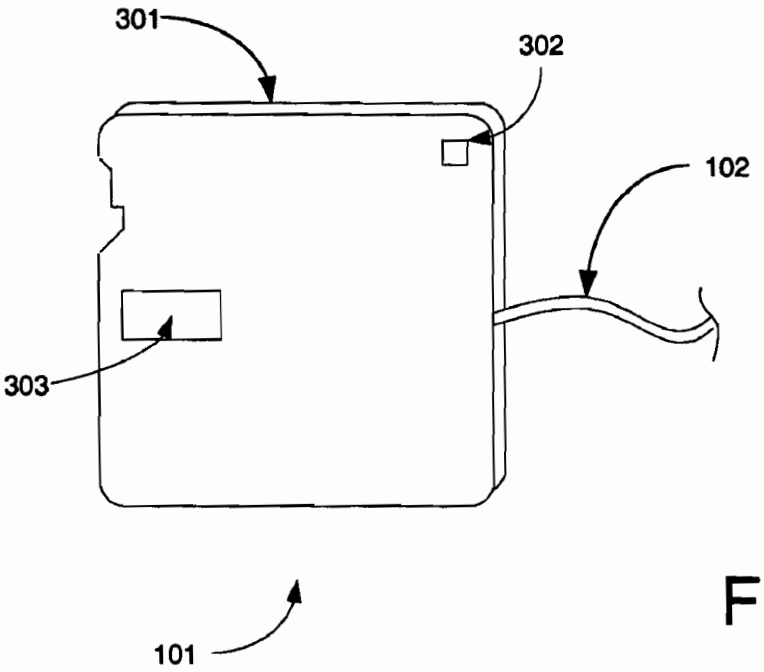


Fig. 3A

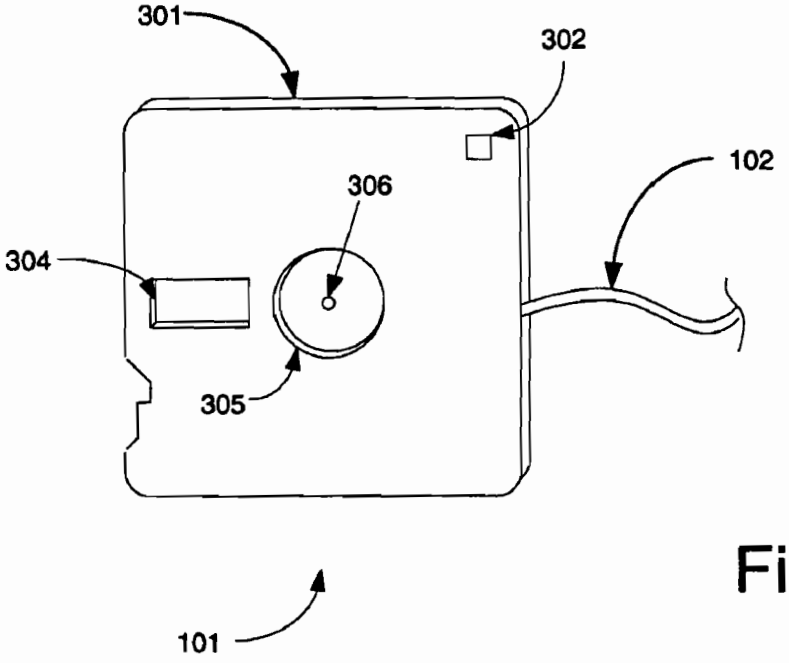


Fig. 3B

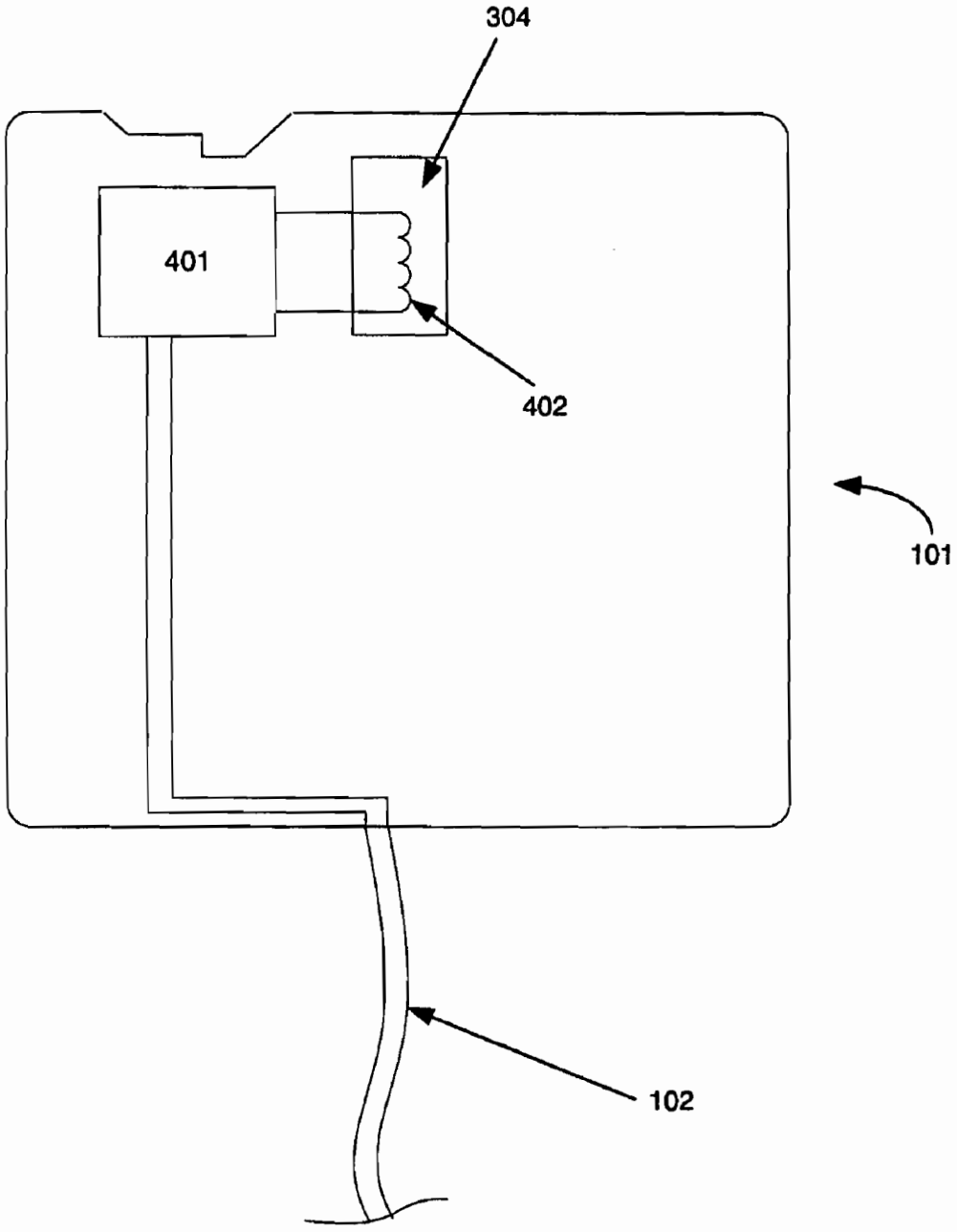


Fig. 4

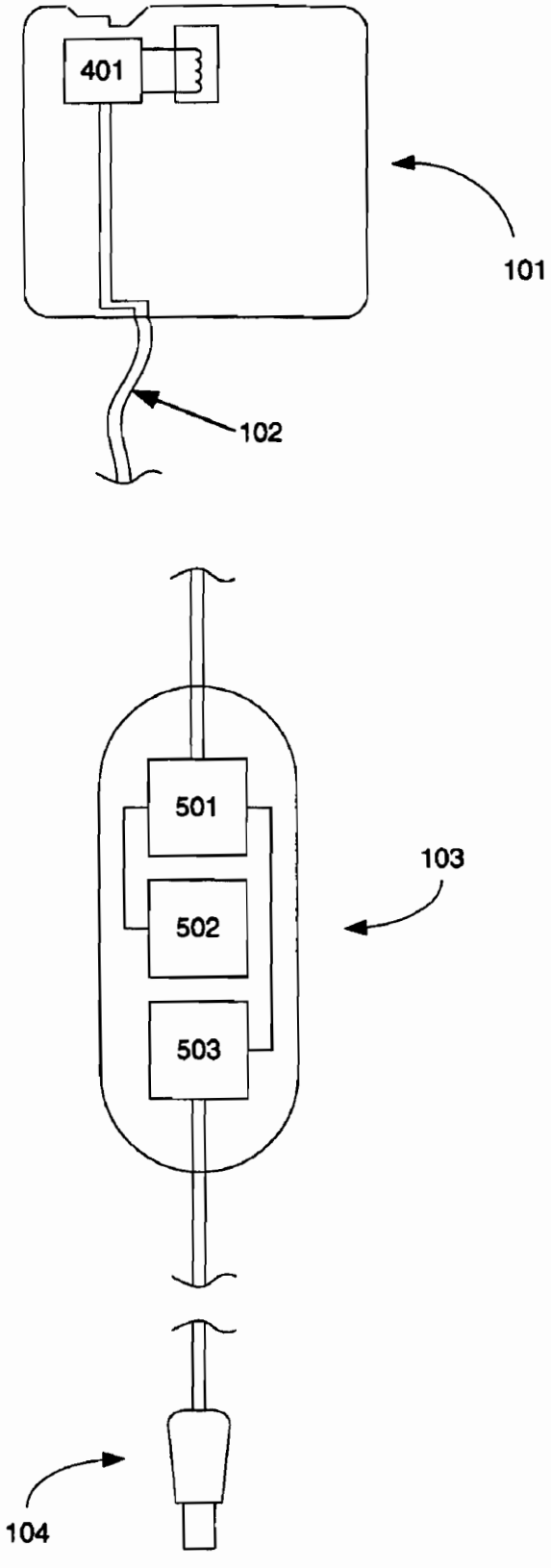
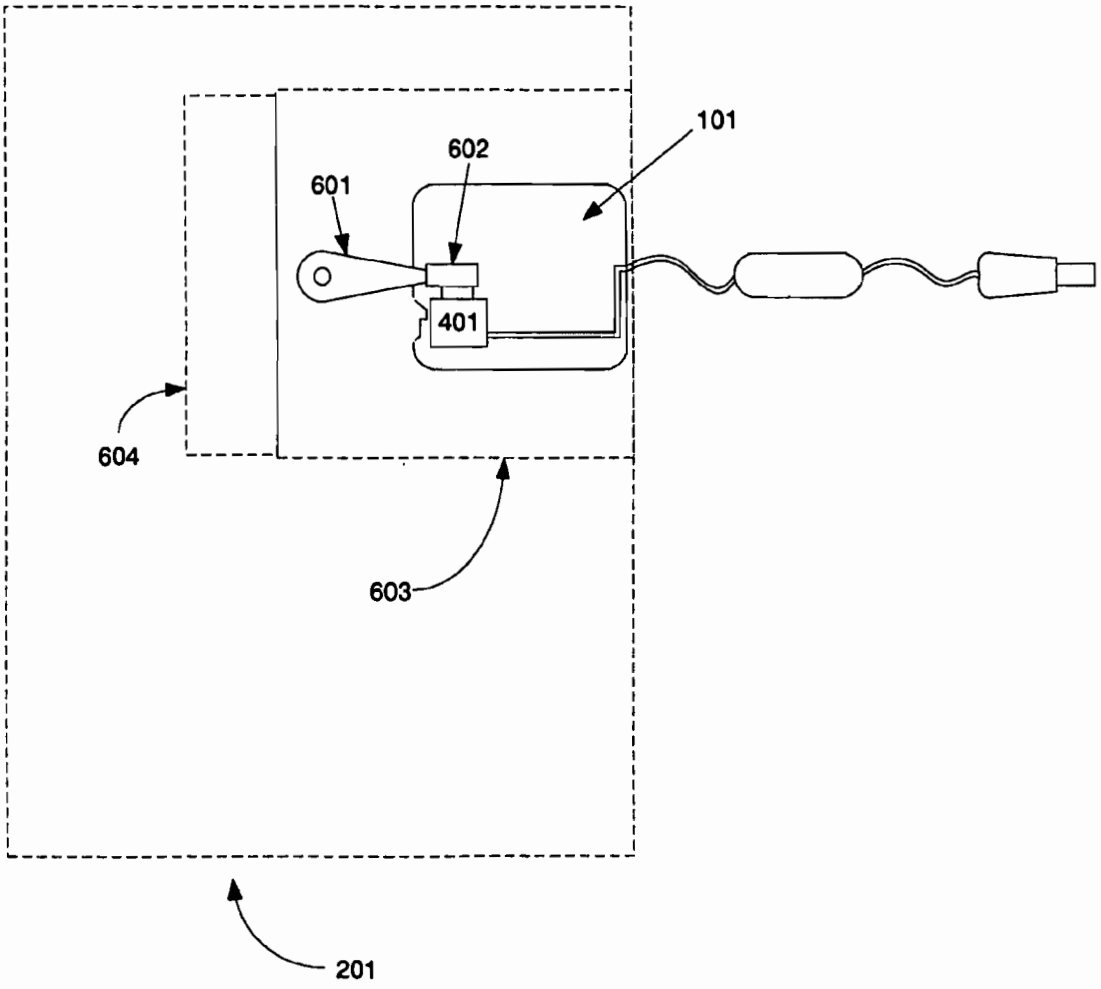


Fig. 5

Fig. 6



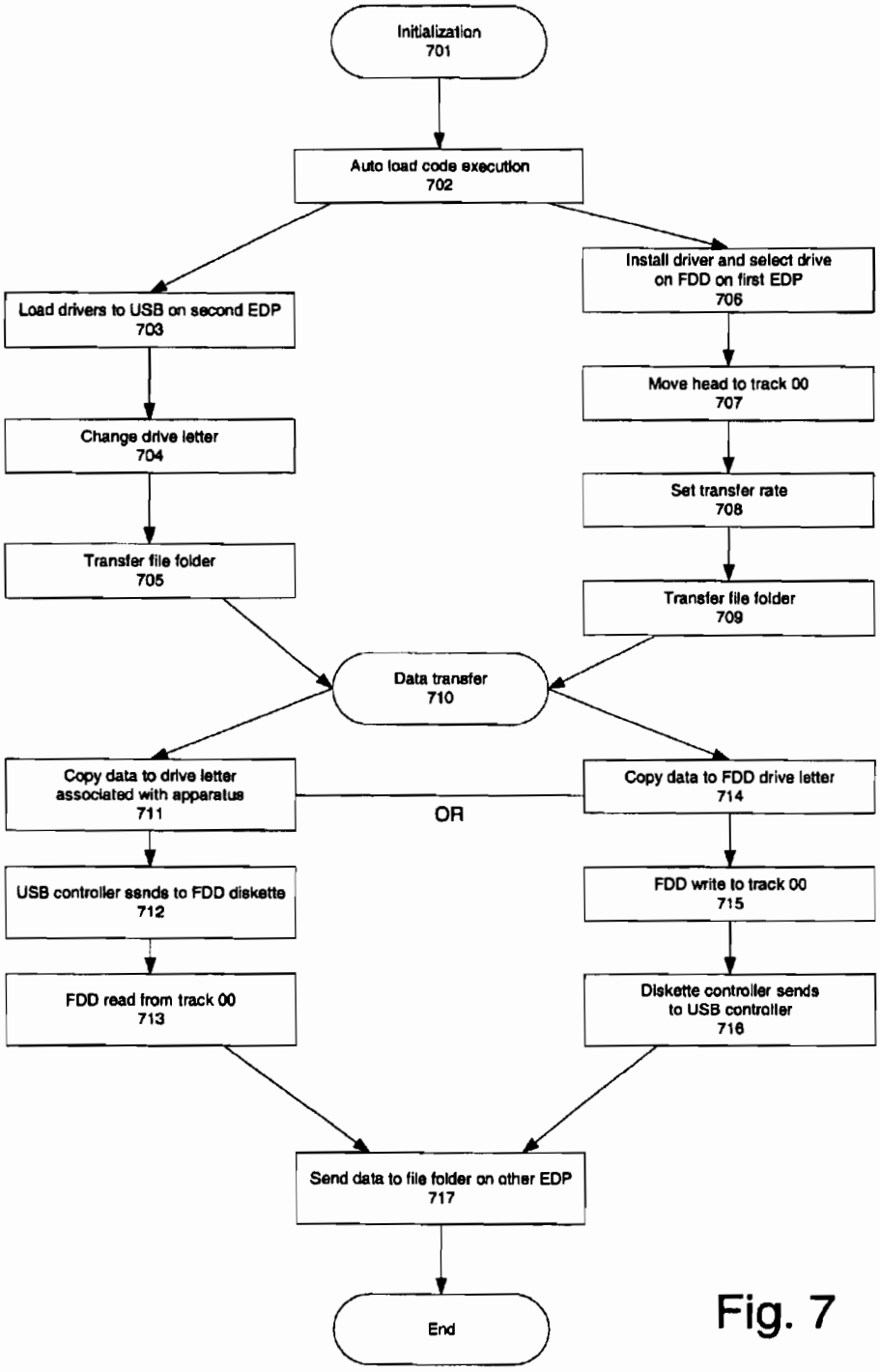

Initialization
701
Auto load code execution
702
Load drivers to USB on second EDP
703
Change drive letter
704
Transfer file folder
705
Install driver and select drive on FDD on first EDP
706
Move head to track 00
707
Set transfer rate
708
Transfer file folder
709
Data transfer
710
Copy data to drive letter associated with apparatus
711
OR
Copy data to FDD drive letter
714
USB controller ssnds to FDD diskette
712
FDD read from track 00
713
FDD write to track 00
715
Diskette controller sends to USB controller
716
Send data to file folder on other EDP
717
End

Fig. 7

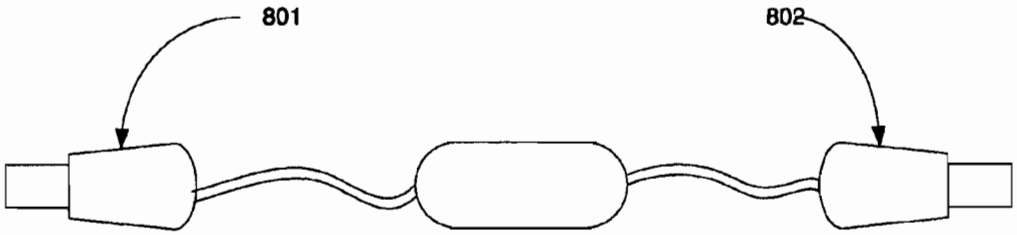


Fig. 8A

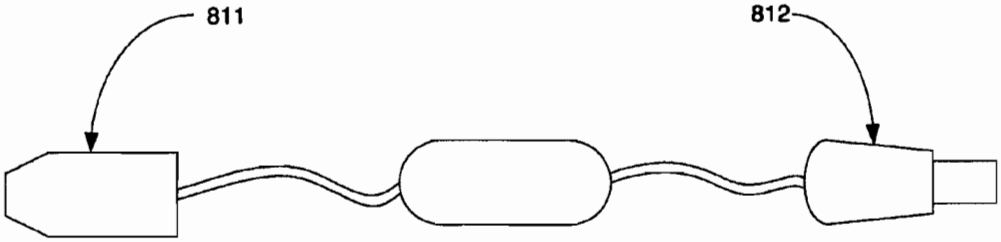


Fig. 8B

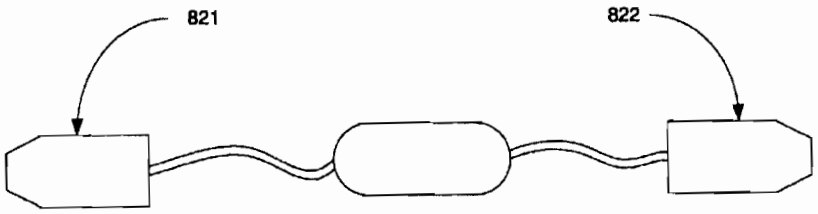


Fig. 8C

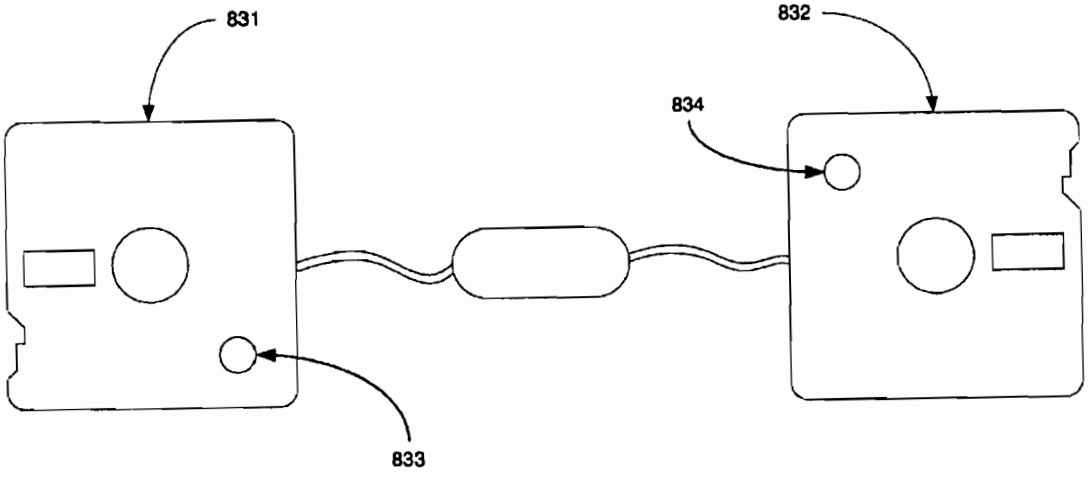


Fig. 8D

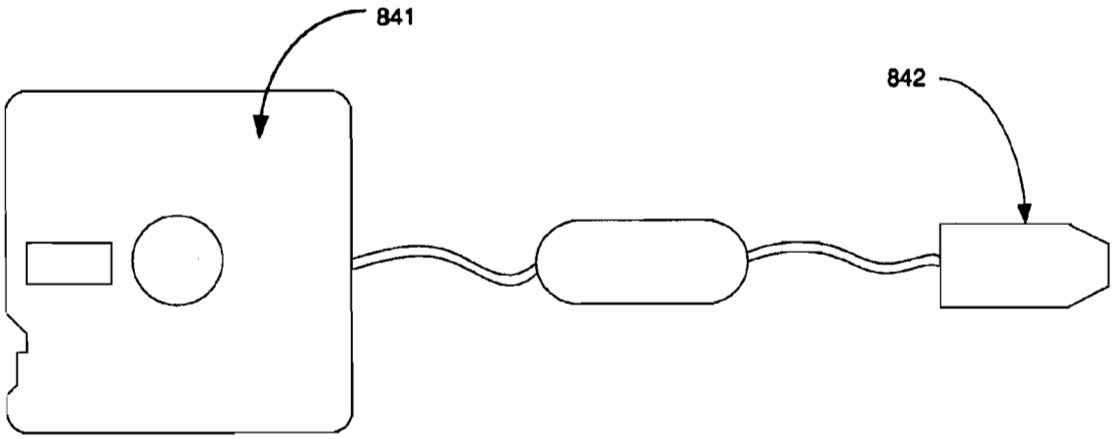


Fig. 8E

INTELLIGENT COMPUTER CABLING

TECHNICAL FIELD

The invention relates generally to the field of data transfer devices, which create a data link between two electronic data processing (EDPs) machines or devices using standard EDP interfaces. More specifically, the invention describes a cable based data transfer system with embedded code to automate the process of moving the data from one EDP to another using standard EDP connectivity interfaces.

BACKGROUND OF THE INVENTION

There are numerous methods of transferring data from one electronic data processing machine (EDP) to another, including copying data to floppy disks, compact disks (CD), flash memory sticks or external data storage devices. There are also software programs and devices available to manage the data transfer using a cable or wireless connection using a standard parallel port, serial port, USB, PCMCI or other network (Ethernet or telephony) interface. These methods require the creation and management of a network.

Almost all of the above methods require manual installation and configuration of the device or the program managing the data transfer, except for the copy function of data to or from a data storage disk using a standard EDP read/write device such as a floppy disk drive (FDD).

The drawback with current cable and wireless methods is that the expertise required to install and configure the device and the related software application to manage the device and execute the desired data transfer is far beyond the expertise of the average computer user. In particular, these prior art data transfer systems lack a process to automate the loading, execution and configuration of the necessary code to facilitate the data transfer between two EDPs.

Therefore, it would be desirable to have an apparatus that automatically loads the drivers and code necessary to facilitate the transfer of data between EDP using standard EDP connectivity interfaces.

SUMMARY OF THE INVENTION

The present invention provides a data transfer system apparatus that automatically loads the necessary drivers and code through two interface elements to facilitate the transfer of electronic data from one EDP to another. In a preferred embodiment of the present invention the apparatus consists of a cable, USB interface plug, FDD transfer device (diskette), processor, controller, memory, circuitry components and software code. The electronic components and software applications are contained in the cable housing unit. Each interface element is attached at one end of the cable so they can be inserted into the respective EDP interfaces. Insertion of the apparatus into the EDP interfaces automatically triggers the execution of the embedded software for auto loading of the necessary code to control the transfer of the data directly from one EDP to the other. The system emulates the apparatus as a peripheral device attached to the EDP through its USB port interface coupled to the other EDP using its FDD to transfer the data, using the data storage capacity of the receiving EDP as the serial bus end-point.

The invention provides an apparatus with an embedded system, which uses flash memory to automate code loading and file execution. This method replaces current data transfer methods between two EDPs that require three separate physical components of cable, software and a peripheral device (or device emulation). The manual loading of software onto each EDP is eliminated by using programmable memory arrays (flash memory) and by using the power source provided by the USB port on one EDP to supply current to the processor(s) and memory.

The present invention allows for a reduction in the steps required to use a cable based data transfer system. Utilization of the FDD allows data transfer to or from EDPs that do not have USB ports, which is helpful when transferring data files from older EDPs. The present invention also eliminates the complexity of manual software application loads and configuration, which provides a low cost data transfer system that can be used by the average non-expert user. Because of the current supplied by the USB port, there is no requirement for an external power source, internal batteries or internal current generator, further reducing the cost of using the invention. Furthermore, the present invention is operating system (OS) agnostic, and the data transfer volumes are limited only by the available data storage capacity of the EDP receiving the transferred data.

The functional result of the apparatus use is an easy-to-use true "plug and play" data transfer system through the emulation of the target EDP as a peripheral storage device connected to the source EDP.

BRIEF DESCRIPTION OF THE DRAWINGS

The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, however, as well as a preferred mode of use, further objects and advantages thereof, will best be understood by reference to the following detailed description of an illustrative embodiment when read in conjunction with the accompanying drawings, wherein:

FIG. **1** depicts a 3.5" FDD compatible diskette in accordance with the present invention;

FIG. **2** shows two EDPs connected with a FDD compatible diskette assembly;

FIG. **3**A shows the top side of the FDD diskette interface in accordance with the present invention;

FIG. **3**B shows the bottom side of the FDD diskette;

FIG. **4** shows an example configuration of the inside of the FDD diskette;

FIG. **5** show the architecture of the cable-housing unit connected to the diskette at one end and a standard USB plug type A on the other end;

FIG. **6** shows the diskette of the present invention inserted into an EDP **201** through a standard 3.5" FDD external interface;

FIG. **7** is a general flowchart of the auto-load process of the first embodiment of the present invention;

FIG. **8**A shows an alternate embodiment of the present invention with USB plugs at both ends of the cable;

FIG. **8**B shows an embodiment of the present invention with a USB plug at one end of the cable and an IEEE-1394 plug at the other end;

FIG. **8**C shows an embodiment of the present invention with IEEE-1394 plugs at both ends of the cable;

FIG. **8**D shows an embodiment of the present invention with FDD interfaces at both ends of the cable; and

FIG. **8**E shows an embodiment of the present invention with a FDD interface at one end of the cable and an IEEE-1394 plug at the other end.

DETAILED DESCRIPTION OF THE DRAWINGS

The present invention provides a cable based data transfer apparatus that contains embedded electronics using flash memory to automatically load the drivers and code to facilitate the transfer of data utilizing standard electronic data processing (EDP) connectivity interfaces.

Universal serial bus (USB) interfaces are becoming the de facto interface standard for connectivity to peripheral devices and is currently included in the manufacturing of new EDPs. USB specifications provide built-in functionality to make peripheral expansion more user friendly as well as providing a single cable model for connectivity to an EDP. These features include self-identification of USB compliant peripherals, auto mapping of functions to a driver and enabling a peripheral device to be dynamically attachable and re-configurable. The USB specification also includes a data flow model, which provides the architecture to manage data transfer from a host platform to an end-point on a device (pipe). The USB Specification provides requirements for the electrical and physical connection between the peripheral device and the host using the bus. An important feature of the USB interface is that it provides up to 500 milliamps of electrical power at 5 volts and signals very fast at 480 Mb/s for high speed USB devices compared to 115 kbits/s for serial and parallel port interfaces.

For the transfer of data from one EDP to another using the USB specification, cables are typically used as the transport medium between a standard USB port on an EDP (connector type A) and a USB compatible peripheral device (connector type B) or another USB port on another EDP. Using the USB specification to transfer data from one EDP to another requires the creation or emulation of a peripheral type device to utilize the embedded USB functionality. This is typically accomplished by loading and configuring a software application that in turn loads the appropriate drivers and provides the necessary code to create the USB end-point and manage what has become a cable based peripheral. This process normally involves loading a compact disk in the CD drive and loading and configuring the necessary application and/or code, which requires considerable expertise on the user's part.

Like USB, IEEE-1394 is an external bus standard that uses twisted pair wiring to move data. It also supplies an electric current along with support for Plug-and-play or "hot plugging" with compatible peripheral devices. The basic feature/functionality sought in the development of this standard is the same as USB, mainly to replace the myriad of I/O connectors employed by consumer electronics equipment and personal computers. Like USB, it supports the concept of an isochronous device, a device that needs a certain amount of bandwidth for streaming data. IEEE-1394 is considered a high performance serial bus in that it supports data transfer rates substantially higher than current USB specifications. It has two forms, 1394a and 1394b with the later supporting transfer rates of 800 Mbps, twice that of 1394a.

IEEE-1394 is a layered transport system. The current standard defines three layers: Physical, Link and Transaction. The Physical layer provides the signals required by the IEEE-1394 bus. The Link layer takes the raw data from the Physical layer and formats it into recognizable 1394 packets. The Transaction layer takes the packets from the Link layer and presents them to the application.

Because of its high data transfer rates and multiplexing capabilities of a variety of different types of digital signals, IEEE-1394 is being adopted as the de facto standard for the transfer of large data volumes, particularly those devices that require real-time transfer of high levels of data such as compressed video and digitized audio. IEEE-1394 interfaces are beginning to be included in the manufacturing of personal EDP machines.

Floppy disk drives (FDDs) have been included in the manufacturing of most EDPs to date. The current standard

for an EDP is an FDD that utilizes a 3.5" floppy magnetic disk. The important feature of a standard FDD relative to this invention is the read/write head, which is used to convert binary data to electromagnetic pulses when writing to the disk, and the reverse when reading from the disk. However, FDDs are being phased out as part of the normal technology life cycle for computer disk drives due to the adoption of the compact disk (CD) and digital versatile disk (DVD).

FDDs are typically used for loading new software applications onto to the memory of the EDP or for extracting data to a floppy disk for storage or data transfer. FDDs are also typically used to create "boot disks" for the EDP's operating system. One of the major drawbacks of FDDs leading to its obsolescence is the limitation of the amount of data that can be stored on a standard floppy disk as well as the slow transfer rates.

Elements exist that can interface with the standard read/write heads of most FDDs using a smart-diskette. This creates a physical transfer interface using a basic magnetic transducer that is essentially a simple antenna-based transmitter and receiver of the electromagnetic pulses created by the FDD's read/write heads. However, these elements lack an automated process and transfer medium for transferring data from one EDP to another. Such smart-diskette based technologies are primarily used to provide an interface for smart cards (e.g., medical patient smart-cards and various peripheral memory cards) to the host EDP through the FDD read/write head mechanism. There are also a number of other drawbacks to current smart-diskette technologies including the requirement for a voltage generator and/or batteries to provide the necessary electrical current to run the necessary processors and controllers and the lack of an interface to any of the current standard EDP interfaces including the USB specification. Other disadvantages include the requirement for loading and configuring a software application prior to usage and the lack of an automated method to self-discover a peripheral plugged into a smart-diskette interface or plug.

Flash-memory using programmable gate array based memory modules is a relatively new type of solid-state technology. This type of electronic non-volatile memory chip can also be erasable. Inside the flash memory chip is a grid of columns and rows, with a two-transistor cell at each intersecting point on the grid. A thin oxide layer separates the two transistors. One of the transistors is known as the floating gate, and the other one is the control gate. The electrons in the cells of a flash-memory chip can be manipulated by the application of an electric field, a higher-voltage charge. Flash-memory uses in-circuit wiring to apply this electric field either to the entire chip or to predetermined sections known as blocks. These blocks can be programmed or erased and re-written. Flash memory works much faster than traditional electrically erasable programmable read-only memory (EEPROM) chips because instead of erasing one byte at a time, flash memory erases a block or the entire chip.

Peripheral devices containing flash memory modules have the advantage of being relatively inexpensive and require relatively little power as compared to traditional magnetic storage disks. Most devices containing flash memory connect to the host EDP using one of the standard EDP interfaces (e.g., USB, PCMCIA, etc.) and then use the low cost chips to either provide a self-contained data storage medium or send a driver to the host EDP and rely on a separately loaded software application to manage the device.

With reference now to the figures, FIG. 1 depicts a 3.5" FDD compatible diskette in accordance with the present

invention. In this embodiment of the invention, the data transfer apparatus **100** comprises a 3.5" FDD compatible diskette **101** containing electronic components connected to a twisted pair cable **102** that is in turn connected to a cable housing unit **103**. The cable housing unit **103** contains additional electronic components mounted on a solid-state board/card and is connected by the twisted pair cable **102** to a USB type A plug **104**.

FIG. **2** shows two EDPs connected by a FDD compatible diskette assembly. The diskette **101** is inserted into the 3.5" FDD **210** of the first EDP **201**, and the USB plug **104** is inserted into the USB port interface **220** of the second EDP **202**. The USB interface, through existing USB specifications and functionality provided with EDP **202**, provides an electrical current to the apparatus **100**. Electrical current is also provided by the twisted pair cable **102** to the diskette **101** to power its electronic components.

When the data transfer apparatus **100** is plugged into the port interface **220** in the second EDP **202**, USB interfaces auto-generate a request signal from the EDP **202**. The processor and flash memory contained in the cable housing unit **103** answers the request from the EDP **202** with a reply that loads the necessary driver(s) and identifies the apparatus **100** as a peripheral storage type device and displays a drive letter and identifier in the EDP operating system's (OS) user interface. The processor in the cable-housing unit **103** then sends a storage file folder to the OS file structure and displays it in the user interface of the OS of EDP **202**.

Simultaneous to the auto-loading of driver(s) and code to EDP **202**, the processor and flash memory in cable housing unit **103** signals the controller **303** in the diskette **101** (shown in FIG. **4**) to initiate the auto load process of drive selection, head alignment to track **00**, and setting of the transfer rate with the FDD **210** of the first EDP **201**. The processor in the cable housing unit **103** then sends a storage file folder to the OS file structure of EDP **201** through the twisted pair cable **102** and the electronic components in the diskette **101** and displays the file in the OS user interface of EDP **201**.

The transfer of data from the first EDP **201** to the second EDP **202** is accomplished by simply copying the desired data to the appropriate FDD drive letter (usually Drive A:) through the default OS user interface resident on EDP **201**. The data flow is regulated by the FDD **210** internal to EDP **201** and controller **303** in diskette **101** to move through the twisted pair cable **102** into the electronic components in cable housing unit **103** and then through twisted pair cable **102** and USB plug **104** into USB port interface **220** in EDP **202**. The USB controller in housing unit **103** manages the flow of the data to EDP **202**, directing it to the loaded file folder.

Transfer rates are dependent on the form implemented including the length and quality of twisted pair cable **102**, its insulation/sheathing qualities, processing speeds of EDP internal processing chips, electrical current strength from USB port **220**, as well as electronic component configurations and module types in cable housing unit **103** and diskette **101**.

With reference now to FIG. **3**A, the top side of the diskette **101** is depicted in accordance with the present invention. The diskette **101** is comprised of an outer casing **301** protecting the electronic components and wiring, which are contained inside the diskette and mounted on a solid-state circuit-type card wired to the twisted pair cable **102**. The diskette **101** is approximately the same width (maybe slightly wider) and length of a standard 3.5" floppy disk. The positioning of the attachment of twisted pair cable **102** can vary depending on the form of the configuration of the inner electronic components and wiring of the inside circuitry board of the diskette.

The write-protect window **302** is the same size and shape and in the same position as write-protect windows found on standard 3.5" floppy disks. The write-protect window **302** is in the open position and contains no moving window or slider so that the diskette emulates a write-ready floppy disk.

The outer casing **301** of diskette **101** also has a cutout **303** on the top of the diskette exposing the inside of the diskette casing. Cutout **303** provides an area where the top read/write head rests while the diskette **101** is in the inserted position inside the FDD.

FIG. **3**B depicts the bottom side of the diskette. A recess **304** accommodates and aligns the bottom read/write head of the FDD. In the center of the diskette **101** there is a circular recess **305** where the drive for a magnetic floppy disk would normally be, with another smaller and deeper circular recess **306** in the center to accommodate the drive spindle of the FDD. The positioning, shape and size of recesses **305**, **306** is the same as found on standard 3.5" floppy disks.

FIG. **4** shows an example configuration of the inside of diskette **101** in accordance with the present invention. Twisted pair cable **102** is wired to a circuitry-type board, which connects the twisted pair wires to the controller **401**. Controller **401** manages the data flow to and from the cable housing unit through twisted pair wires **102**. The controller **401** also controls data flow to and from the FDD by means of an electrically connected magnetic transducer **402** that receives and sends the signal pulses to and from the read/write head of the FDD. The read/write head sits in recess **304** to align the head on the diskette **101** so that an emulation of a 3.5" floppy disk set at track 00 can be accomplished using the magnetic transducer **402** as an antenna-type receiver/transmitter of the electromagnetic pulse signals.

FIG. **5** shows the architecture of the cable-housing unit connected to the diskette at one end and a standard USB plug type A on the other end. The cable-housing unit **103** contains a solid-state circuit-type board/card configuration holding a microprocessor **501**, memory (flash-type) **502** and a USB controller **503** along with wiring connecting the board and electrical components to the twisted pair cable **102**. Processor **501** is connected to the circuitry-type board allowing it to send and receive signals to and from the diskette controller **401** and USB controller **503** as well as receive electrical current from the USB port interface on the EDP. The flash memory **502** module is a floating gate array type module containing all the code necessary to perform the execution of the application loads and driver installations upon system initialization when the apparatus is inserted into the first and second EDPs. The USB controller **503** manages the data flow and interaction with the second EDP using standard USB specifications and functionality, as described above.

FIG. **6** shows the diskette **101** of the present invention inserted into an EDP **201** through a standard 3.5" FDD external interface. The internal interface is depicted by showing diskette **101** in the inserted position and the FDD top arm assembly **601** holding read/write head **602** resting in recess of the diskette. Internal control of the FDD **603** is provided by the disk controller **604**, which manages the data transfer internally between the FDD **603** and the internal processor and memory components of the EDP **201**. These components are found with most all FDD devices.

FIG. **7** is a general flowchart of the auto-load process of the present invention. The process is achieved by executing software code embedded in the memory of the apparatus

contained in cable housing unit. The process begins with insertion of the diskette into the FDD interface of the first EDP and insertion of the USB plug into the USB port interface of the second EDP, which activates the initialization of the auto load process (step **701**). The USB port interface provides the electrical current to the apparatus to power the processor and other electronic components contained in the cable housing unit and diskette. Software code execution then launches two parallel processes of loading the necessary file(s), driver(s) and code to each EDP (step **702**).

The first process stream begins by answering the request generated by the second EDP and sending a response and the necessary driver(s) identifying the apparatus as a peripheral device (step **703**). The auto-loading of the driver(s) creates a drive letter displayed in the OS user interface of the EDP identifying the apparatus as a peripheral device (step **704**). The apparatus then transfers a file folder to the file structure of the EDP OS and displays it as a file related to the data transfer system apparatus (step **705**).

The second process stream begins by installing a driver on the first EDP and sending a signal to the FDD identifying the diskette as a drive, using the default OS identifier for the FDD (normally displayed as drive A: in most operating systems) (step **706**). The apparatus then sends a signal to the FDD disk controller to move the read/write head to track **00** (step **707**). The diskette controller accommodates the emulation of the diskette as a floppy disk with track **00**. The data transfer rate is set in the same manner of sending a signal managed by the controller through the magnetic transducer to the read/write head of the FDD (step **708**). The apparatus then auto transfers a file folder to the file structure of the first EDP OS and displays it as a file related to the data transfer system apparatus (step **709**).

The data transfer process can now begin on each EDP by using the existing OS user interface of each machine to copy and move the files from one machine to another (step **710**).

To copy data from the second EDP to the first, the user copies the data to the drive letter (i.e. A:) that identifies the drive as the apparatus (step **711**). The copy procedure is the same procedure already used by the user to copy data and files from one location to another using the character based command line user interface or the graphical user interface (GUI) provided by the EDP's OS. When the copy function is completed, the USB controller sends the data to the cable-housing unit, which passes the data to diskette controller, and the diskette controller then sends the data as signals to the read/write head as an emulation of track **00** on a floppy disk (step **712**). The FDD of the first EDP reads from track **00** (step **713**) and sends the data to the file folder that was sent to the first EDP in step **709** earlier in the auto load process (step **717**).

Transfer of data from the first EDP to the second is essentially the reverse of steps **711–713**. The process begins by copying the desired data from the first EDP to the FDD drive letter (step **714**). Again, the copy procedure is the same procedure typically used to copy data and files from one location to another. When the copy function is completed, the FDD disk controller writes the data to track **00** (step **715**), which is then picked up by the magnetic transducer and sent by the diskette controller to the USB controller through the cable-housing unit (step **716**). The data transfer process is completed by the USB controller sending the data through the USB port interface to the file folder on the second EDP (step **717**).

In both copy processes, the users of the EDPs use the existing user interfaces of their respective machines provided by the operating systems. The default copy, move, and erase procedures are also followed to move the transferred data from the storage file folder placed in the EDPs' file structure in step **704** and **709** to the desired location on the EDPs. Using the present invention, the data volume that can be transferred from one EDP to another is limited only by the total available data storage capacity of the EDP receiving the transferred data.

In addition to the example embodiment described above employing 3.5" FDD and USB interfaces, the present invention may also be implemented with the IEEE-1394 standard. By incorporating the FDD, USB and IEEE-1394 interfaces, the present invention is capable of five alternate embodiments in addition to the one described above.

FIG. **8**A shows an alternate embodiment of the present invention with USB plugs **801, 802** at both ends of the cable.

FIG. **8**B shows an embodiment of the present invention with a USB plug **811** at one end of the cable and an IEEE-1394 plug **812** at the other end.

FIG. **8**C shows an embodiment of the present invention with IEEE-1394 plugs **821, 822** at both ends of the cable.

FIG. **8**D shows an embodiment of the present invention with FDD interfaces **831, 832** at both ends of the cable using a battery **833, 834** inserted into each diskette to provide the necessary current to power the controller.

FIG. **8**E shows an embodiment of the present invention with a FDD interface **841** at one end of the cable and an IEEE-1394 plug **842** at the other end.

The USB and IEEE-1394 interfaces provide almost identical feature/functionality in terms of issuing and handling requests from a peripheral device. (The invention apparatus is emulating a peripheral storage device.) USB and IEEE-1394 specifications are managed by separate governing bodies but the way in which the invention sends and receives data using the cable-based system is the same. The embodiments that include an FDD interfaces are more complicated than the USB and IEEE-1394 ones in that additional electronics are required to transfer, manage and control the data through the read/write head of the FDD. However, because the additional electronics are contained inside the diskette unit itself a single cable-housing unit can be manufactured to support all six embodiments. In this way, only the interface plugs/devices at the end of the cable change, which significantly reduces the cost to manufacture multiple products that have the same end function and user experience.

The description of the present invention has been presented for purposes of illustration and description, and is not intended to be exhaustive or limited to the invention in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art. The embodiment was chosen and described in order to best explain the principles of the invention, the practical application, and to enable others of ordinary skill in the art to understand the invention for various embodiments with various modifications as are suited to the particular use contemplated. It will be understood by one of ordinary skill in the art that numerous variations will be possible to the disclosed embodiments without going outside the scope of the invention as disclosed in the claims.

I claim:

1. An apparatus for data transfer between two electronic data processing (EDP) devices, the apparatus comprising:

(a) a cable housing;

(b) a cable extending from two points of said cable housing;

(c) a solid state board inside said cable housing, wherein the solid state board is wired to said cable;

(d) a processor and memory chip mounted on said solid state board;

(e) a first EDP interface at the first end of said cable; and

(f) a second EDP interface at the second end of said cable;

wherein, upon insertion of the first EDP interface into the first EDP device and insertion of the second EDP interface into the second EDP device, the apparatus automatically loads and executes software code stored in said memory chip onto said EDP devices, wherein said software code controls the direct transfer and storage of data from one EDP device to the other EDP device.

2. The apparatus according to claim **1**, further comprising a controller mounted on said solid state board.

3. The apparatus according to claim **1**, wherein the memory chip is flash memory.

4. The apparatus according to claim **1**, wherein the apparatus emulates a peripheral storage device.

5. The apparatus according to claim **1**, wherein transfer of selected data between the EDP devices is performed using respective existing operating systems and user interfaces of each EDP device.

6. The apparatus according to claim **1**, wherein the first EDP interface may be one of the following:

a universal serial bus plug;

an IEEE-1394 plug;

a diskette compatible with a 3.5 inch floppy disk drive, wherein the diskette contains a controller on a circuitry board wired to said cable and a magnetic transducer connected to the controller that transfers data through the read/write head of the floppy disk drive.

7. The apparatus according to claim **1**, wherein the second EDP interface may be one of the following:

a universal serial bus plug;

an IEEE-1394 plug;

a diskette compatible with a 3.5 inch floppy disk drive, wherein the diskette contains a controller on a circuitry board wired to said cable and a magnetic transducer connected to the controller that transfers data through the read/write head of the floppy disk drive.

8. A method for transferring data between two electronic data processing (EDP) devices, the method comprising the steps of:

(a) inserting a first EDP interface into the first EDP device and inserting a second EDP interface into the second EDP, wherein the first and second EDP interfaces are connected by a cable that extends from two points of a cable housing, wherein said cable housing contains a solid state board wired to said cable, and wherein a processor and memory chip are mounted on said solid state board; and

(b) automatically loading and executing software code stored in said memory chip onto said first and second EDP devices upon insertion of the first EDP interface into the first EDP device and insertion of the second EDP interface into the second EDP device, wherein said software code controls the direct transfer and storage of data from one EDP device to the other EDP device.

9. The method according to claim **8**, wherein step (b) further comprises automatically selecting a drive on both the first EDP device and second EDP device, wherein said selected drives are used for sending and receiving data.

10. The method according to claim **8**, wherein at least one of the first and second EDP interfaces transfers data through the read/write head of a 3.5 inch floppy disk drive.

11. The method according to claim **10**, wherein the read/write heads is automatically set to read from track **00**.

12. The method according to claim **8**, wherein step (b) further comprises transferring a storage file from one EDP device to the file directory of a resident operating system in the other EDP device.

13. The method according to claim **8**, wherein the transfer of data between the two EDP devices may be both unidirectional and bidirectional.

14. The method according to claim **8**, wherein the device comprising the first and second EDP interfaces, cable, and cable housing emulates a peripheral storage device.

15. The method according to claim **8**, wherein transfer of selected data between the EDP devices is performed using respective existing operating systems and user interfaces of each EDP device.

* * * * *